New features in PFIM for optimal design in nonlinear-mixed effects models using the R package PFIM 5.0

Romain Leroux, Jérémy Seurat, Hervé Le Nagard,
NGUYỄN Trần Bách, France Mentré on behalf of the PFIM group

IAME, UMR 1137, INSERM, Université Paris Cité, Paris, France

Background and objectives

- Nonlinear mixed-effects models (NLMEMs) are widely used in model-based drug development to analyze longitudinal data. For optimizing the design of longitudinal studies in pharmacometrics, the use of the Fisher information matrix (FIM) is a good alternative to time-consuming clinical trial simulations
- PFIM 4.0 was released in 2014 [1] and is one of the tools developed for FIM-based evaluation and/or optimization of designs in NLMEMs
- The R package PFIM 5.0 is the new version of PFIM, that provides powerful tools to perform FIM evaluation and design optimization under given design constraints in NLMEMs
- PFIM 5.0 is based on the object-oriented programming language S4

The R package PFIM 5.0

Methods

- PFIM 5.0 was re-implemented from scratch in R S4 language, following a top-down approach [2]
- The object-oriented system S4 defines objects having clear object-oriented programming characteristics including class and argument definitions, inheritance, as well as argument checking, instantiation and implementation methods
- The user can write his entire project in a single R script
- PFIM 5.0 can be downloaded on the Comprehensive R Archive Network at **https://cran.rproject.org/web/packages/PFIM/index.html**
- Documentation with additional detailed examples is provided by a comprehensive user guide from the PFIM website at **http://www.pfim.biostat.fr/**

Notable features

- Individual, Population and Bayesian Fisher Information Matrix
- User-defined models (analytic and ODE) for one or several responses
- Library of PK, PD and PKPD models: the PK library includes different administration (bolus, infusion and oral - first order absorption), linear and Michaelis-Menten elimination, 1- and 2-compartment models. The PD library contains direct and indirect, linear and nonlinear models. The PK/PD models are obtained by combining the models from the PK and PD libraries. PFIM 5.0 also offers the possibility to extend models from the library
- Parameters with different distributions (normal and lognormal)

New features

- Eased definition of multiple administrations with different routes (oral, IV bolus, IV infusion) and at different doses
- PKPD and PKPKPD models in analytic and ODE form
- A data summary for design evaluation and optimization. The standard data visualization package ggplot2 is used to display the results in clear graphical form (sensitivity graphs, responses over time and optimal design, SE and RSE obtained with the evaluated or the optimised design)
- The package rmarkdown is used to turn all the results into high quality reports that can be easily shared
- Possibility to get and use the FIM based results after design evaluation or optimization
- Optimization algorithms: the simplex algorithm (Nelder-Mead) [3], PGBO (Population Genetic Based Optimization) [4], PSO (Particle Swarm Optimization) [5], Fedorov-Wynn [6], Multiplicative algorithm [7,8]
- Possibility to optimize both the doses and the measurement times (e.g. by using the Multiplicative algorithm)
- Developer documentation on all the methods and classes implemented
- User-friendly vignettes to demo of the package's capabilities

Perspectives

- Covariates and Wald test power predictions [9]
- Alternative methods to evaluate the FIM (e.g. MC/AGQ [10]) for discrete response models and robust design optimization accounting for model and/or parameter uncertainty [11,12]
- Increase the interoperability of PFIM and the library of PKPD model with estimation parameter softwares
- A GUI version of the package PFIM 5.0

Example of design evaluation and optimization of an ODE PKPD model

R script

```
# Design evaluation

### Create two PFIM projects:
### - `MyProject_evaluation`: project for the design evaluation named « eval_PKPD_FloresMurrieta1998 »
### - `MyProject_optimization`: project for the design optimization named « opti_PKPD_FloresMurrieta1998 »

MyProject_evaluation = PFIMProject( name = "eval_PKPD_FloresMurrieta1998" )
MyProject_optimization = PFIMProject( name = "opti_PKPD_FloresMurrieta1998" )

### Create the statistical model and set the parameters for the ode solver

MyStatisticalModel = StatisticalModel()

### Define the model equations of the PKPD model

MyModelEquations  =  ModelODEquations( list(RespPK = expression( Cc ),
                                            RespPD = expression( E )) ,
                                       list("Deriv_Cc" = expression( dose_RespPK/V*ka*exp(-ka*t) - Cl/V*Cc ),
                                            "Deriv_E" = expression(Rin*(1-Imax*(Cc**gamma)/(Cc**gamma +
                                                                    IC50**gamma))-kout*E) ) )
### Assign the PKPD model the statistical model

MyStatisticalModel = defineModelEquations( MyStatisticalModel, MyModelEquations )

### Create the variables of the PKPD model

vCc  =  ModelVariable( "Cc" )
vE  =  ModelVariable( "E" )

### Assign the model variables to the statistical model

MyStatisticalModel  =  defineVariable( MyStatisticalModel, vCc )
MyStatisticalModel  =  defineVariable( MyStatisticalModel, vE )

### Set mu and omega for each parameter

pV = ModelParameter( "V", mu = 0.74, omega = 0.316, distribution = LogNormalDistribution() )
pCl = ModelParameter( "Cl", mu = 0.28, omega = 0.456, distribution = LogNormalDistribution() )
pka = ModelParameter( "ka", mu = 10, fixedMu = TRUE, omega = sqrt( 0 ), distribution = LogNormalDistribution() )
pkout = ModelParameter( "kout", mu = 6.14, omega = 0.947, distribution = LogNormalDistribution() )
pRin = ModelParameter( "Rin", mu = 614, fixedMu = TRUE, omega = sqrt( 0 ), distribution = LogNormalDistribution() )
pImax = ModelParameter( "Imax", mu = 0.76, omega = 0.439, distribution = LogNormalDistribution() )
pIC50 = ModelParameter( "IC50", mu = 9.22, omega = 0.452, distribution = LogNormalDistribution() )
pgamma = ModelParameter( "gamma", mu = 2.77, omega = 1.761, distribution = LogNormalDistribution() )

### Assign the model parameters to the statistical model

MyStatisticalModel = defineParameter( MyStatisticalModel, pka )
MyStatisticalModel = defineParameter( MyStatisticalModel, pV )
MyStatisticalModel = defineParameter( MyStatisticalModel, pCl )
MyStatisticalModel = defineParameter( MyStatisticalModel, pkout )
MyStatisticalModel = defineParameter( MyStatisticalModel, pRin )
MyStatisticalModel = defineParameter( MyStatisticalModel, pImax )
MyStatisticalModel = defineParameter( MyStatisticalModel, pIC50 )
MyStatisticalModel = defineParameter( MyStatisticalModel, pgamma )

### Create and add the error model to the responses PK and PD. Create and add the responses PK and PD to the statistical model

MyStatisticalModel = addResponse( MyStatisticalModel, Response( "RespPK", Proportional( sigma_slope = 0.21 ) ) )
MyStatisticalModel = addResponse( MyStatisticalModel, Response( "RespPD", Constant( sigma_inter = 9.6 ) ) )

### Assign the statistical model to the PFIM projects

MyProject_evaluation = defineStatisticalModel( MyProject_evaluation, MyStatisticalModel )
MyProject_optimization = defineStatisticalModel( MyProject_optimization, MyStatisticalModel )

### Create a design called `MyDesign`

MyDesign =  Design( name = "MyDesign")

### Define the arms and for each arm
    - create and add the administration parameters for the response PK
    - create and add the sampling times for the responses PK and PD

brasTest1 = Arm( name="0.2mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest1 = addAdministration( brasTest1, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(0.2) ) )
brasTest1 = addSampling( brasTest1, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest1 = addSampling( brasTest1, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

brasTest2 = Arm( name="0.64mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest2 = addAdministration( brasTest2, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(0.64) ) )
brasTest2 = addSampling( brasTest2, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest2 = addSampling( brasTest2, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

brasTest3 = Arm( name="2mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest3 = addAdministration( brasTest3, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(2) ) )
brasTest3 = addSampling( brasTest3, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest3 = addSampling( brasTest3, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

brasTest4 = Arm( name="6.24mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest4 = addAdministration( brasTest4, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(6.24) ) )
brasTest4 = addSampling( brasTest4, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest4 = addSampling( brasTest4, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

brasTest5 = Arm( name="11.24mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest5 = addAdministration( brasTest5, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(11.24) ) )
brasTest5 = addSampling( brasTest5, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest5 = addSampling( brasTest5, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

brasTest6 = Arm( name="20mg Arm", arm_size = 6, cond_init = list("Cc"=0,"E"=100) )
brasTest6 = addAdministration( brasTest6, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(20) ) )
brasTest6 = addSampling( brasTest6, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )
brasTest6 = addSampling( brasTest6, SamplingTimes( outcome = "RespPD", sample_time = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) ) )

### Add the arms to the design `MyDesign`

MyDesign  =  addArm( MyDesign, brasTest1 )
MyDesign  =  addArm( MyDesign, brasTest2 )
MyDesign  =  addArm( MyDesign, brasTest3 )
MyDesign  =  addArm( MyDesign, brasTest4 )
MyDesign  =  addArm( MyDesign, brasTest5 )
MyDesign  =  addArm( MyDesign, brasTest6 )

### Add the design `MyDesign` to the PFIM project `MyProject_evaluation`

MyProject_evaluation  =  addDesign( MyProject_evaluation, MyDesign )

### Evaluate the population FIM

evaluationPop  =  EvaluatePopulationFIM( MyProject_evaluation )

### Display the results of the design evaluation

show( evaluationPop )

### Create and save the report for the design evaluation

outputPath = ".....«  # set the path and name of the report to save the report
plotOptions = list( unitTime=c("hour"), unitResponses= c("mcg/mL","DI%") )
evaluationPop = setNamePFIMProject( evaluationPop, "PKPD_FloresMurrieta1998_populationFIM" )

reportPFIMProject( evaluationPop,
                   outputPath = outputPath, plotOptions = plotOptions )

# Design optimization with the Fedorov-Wynn algorithm

### Define the arm and
    - create and add the administration parameters for the response PK
    - create and add the sampling times for the responses PK and PD

brasTest = Arm( name="20mg Arm", arm_size = 30, cond_init = list( "Cc" = 0,"E"= expression( Rin/kout ) ) )
brasTest = addAdministration( brasTest, Administration( outcome = "RespPK", time_dose = c(0), amount_dose = c(20) ) )
brasTest = addSampling( brasTest, SamplingTimes( outcome = "RespPK", sample_time = c( 0.25, 1, 4 ) ) )
brasTest = addSampling( brasTest, SamplingTimes( outcome = "RespPD", sample_time = c( 1.5, 2, 6 ) ) )

### Create and add the design `MyDesign2` to the project `MyProject_optimization`

MyDesign2= Design( name = "MyDesign2")
MyDesign2 = addArm( MyDesign2, brasTest )
MyProject_optimization = addDesign( MyProject_optimization, MyDesign2 )

### Create a sampling constraint for the responses PK and PD

samplingRespPK = SamplingConstraint( response = "RespPK" )
samplingRespPD = SamplingConstraint( response = "RespPD" )

### Define the vector of allowed sampling times for the responses PK and PD

samplingRespPK = allowedDiscretSamplingTimes( samplingRespPK, list( c( 0.25, 0.75, 1, 1.5, 2, 4, 6 ) ) )
samplingRespPD = allowedDiscretSamplingTimes( samplingRespPD, list( c( 0.25, 0.75, 1.5, 2, 3, 6, 8, 12 ) ) )

### Set the initial vectors for the Fedorov-Wynn algorithm

initialElementaryProtocols = list( c( 0.25, 1, 4 ), c( 1.5, 2, 6 ) )

### Set the number of optimisable sampling times

samplingRespPK = numberOfSamplingTimesIsOptimisable( samplingRespPK, c(3) )
samplingRespPD = numberOfSamplingTimesIsOptimisable( samplingRespPD, c(3) )

### Fix certain time values

samplingRespPK = FixTimeValues( samplingRespPK, c( 0.25, 4 ) )
samplingRespPD = FixTimeValues( samplingRespPD, c( 2, 6 ) )

### Create a design constraint and add the sampling constraints to the design

Constr = DesignConstraint()
Constr = addSamplingConstraint( Constr, samplingRespPK )
Constr = addSamplingConstraint( Constr, samplingRespPD )

### Create an administration for response PK

administrationResp = AdministrationConstraint( response = "RespPK" )

### Define the vector of allowed amount of doses for the response PK

administrationResp  =  AllowedDoses( administrationResp, c(20,64) )
Constr = addAdministrationConstraint( Constr, administrationResp )

### Define the total number of individuals to be considered

Constr = setTotalNumberOfIndividuals( Constr, 30 )

### Add the design to the project

MyProject_optimization = setConstraint( MyProject_optimization, Constr )

### Set the vector of initial proportions or numbers of subjects for each elementary design

numberOfSubjects = c(30)
proportionsOfSubjects = c(30)/30

### Set the the parameters of the FedorovWynn algorithm and run the algorithm for the design optimization with a population FIM

optimizer  =  FedorovWynnAlgorithm( initialElementaryProtocols,numberOfSubjects, proportionsOfSubjects, showProcess = T )
optimization_populationFIM  =  OptimizeDesign( MyProject_optimization , optimizer, PopulationFim() )

### Display the results of the design optimization

show( optimization_populationFIM )

### Reports for the design optimization

outputPath = « ... »  # set the path and name of the report to save the report
plotOptions = list( unitTime=c("hour"), unitResponses= c("mcg/mL","DI%") )

reportPFIMProject( optimization_populationFIM, outputPath = outputPath, plotOptions = plotOptions  )
```

Description of the study

Model and data

Taken from a PKPD population study on a non-steroidal molecule [13]

Administration

Single doses of 0.2, 0.64, 2, 6.24, 11.24, or 20mg were given respectively to 6 groups, each of which contained 6 rats, following a parallel design

Evaluation

Blood sampling and drug response evaluation were conducted at 15, 30, and 45min and at 1, 1.25, 1.5, 2, 3 and 4 hours after administration

Optimization

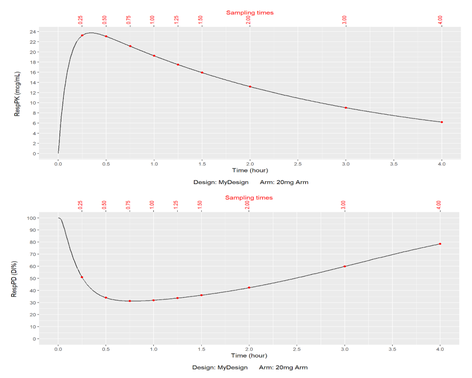
- Algorithm used: Fedorov-Wynn
- Total number of individuals: 30, which can be allocated into different groups (elementary designs)
- Allowed sampling times for the design: 7 for the PK and 8 for the PD, not necessarily the same as those previously defined
- Sampling constraints: 3 sampling times with 2 fixed and 1 optimized, for both the PK and the PD (sampling times can also be different between the PK and PD)
- Administration constraints: 2 possible doses of 20mg and 64mg

Results for design evaluation

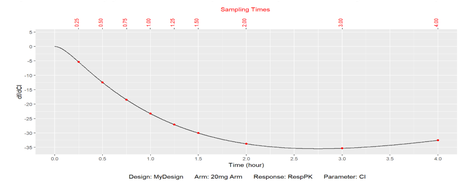
Determinant, condition numbers and D-criteria of the FIM

Determinant	Condition numbers		D-criterion
	Fixed effects	Random effects	
4.154031e+22	4692.284	16696.61	41.2672

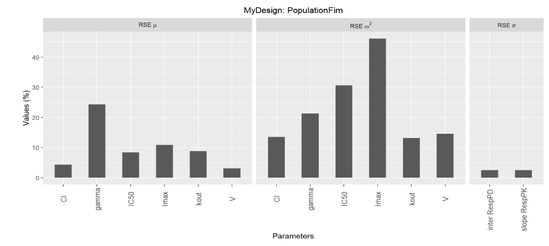
Plots of the evolution of the PK response and PD response for the arm « 20mg Arm »



Plot of the evolution of the sensitivity function with respect to the parameter Cl for the PK response for the arm « 20mg



Plot of the RSE



Results for design optimization with Fedorov-Wynn algorithm

Initial design

Design name	Arms name	Response	Sampling times	Number of subjects
MyDesign2	20mg Arm	RespPK	(0.25, 1, 4)	30
MyDesign2	20mg Arm	RespPD	(1.5, 2, 3)	30

Determinant, condition numbers and D-criteria of the FIM

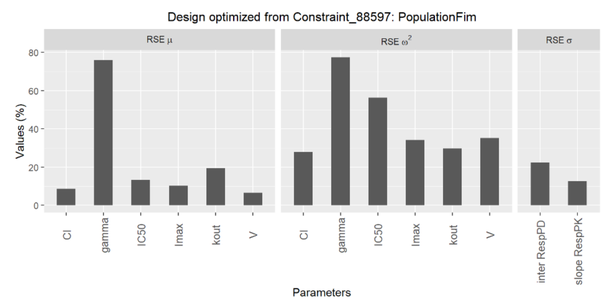
Determinant	Condition numbers		D-criterion
	Fixed effects	Random effects	
5807.961	7.961667e+17	3692.259	1.857201

Optimal design

Arm name	Response	Time dose	Amount dose	Sampling times	Number of subjects
Arm 57	RespPK	0.	20.	(0.25, 1, 4)	5.89779995720863
Arm 57	RespPD			(2, 6, 12)	5.89779995720863
Arm 6	RespPK	0.	64.	(0.25, 0.75, 4)	13.1159366256486
Arm 6	RespPD			(0.75, 2, 6)	13.1159366256486
Arm 60	RespPK	0.	64.	(0.25, 4, 6)	10.9862634171427
Arm 60	RespPD			(2, 6, 12)	10.9862634171427

Determinant, condition numbers and D-criteria of the FIM

Determinant	Condition numbers		D-criterion
	Fixed effects	Random effects	
6.850803e+19	7849.887	10790.31	26.11191



References

[1] Dumont C, Lestini G, Le Nagard H, Mentré F, Comets E, Nguyen TT, et al. PFIM 4.0, an extended R program for design evaluation and optimization in nonlinear mixed-effect models. Comput Methods Programs Biomed. 2018;156:217–29.
[2] Chambers JM. Object-Oriented Programming, Functional Programming and R. Stat Sci. 2014;29:167–80.
[3] Nelder JA, Mead R. "A simplex method for function minimization." Comput J. 1965;7:308–13.
[4] Le Nagard H, Chao L, Tenaillon O. The emergence of complexity and restricted pleiotropy in adapting networks. BMC Evol Biol 2011;11:326.
[5] Eberhart RC, Kennedy J. A new optimizer using particle swarm theory. Proc. of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, 4-6 October 1995;39-43.
[6] Fedorov, VV. Theory of Optimal Experiments. Academic Press, New York, 1972.
[7] Yu Y. Monotonic convergence of a general algorithm for computing optimal designs. Ann Stat. 2010;38:1593–606.
[8] Seurat J, Tang Y, Mentré F, Nguyen, TT. Finding optimal design in nonlinear mixed effect models using multiplicative algorithms. Computer Methods and Programs in Biomedicine. 2021;207.106126.
[9] Retout S, Comets E, Samson A, Mentré F. Design in nonlinear mixed effects models: optimization using the Fedorov-Wynn algorithm and power of the Wald test for binary covariates. Stat Med. 2007;26:5162–79.
[10] Ueckert S, Mentré F. A new method for evaluation of the Fisher information matrix for discrete mixed effect models using Monte Carlo sampling and adaptive Gaussian quadrature. Comput Stat Data Anal. 2017;111:203–19.
[11] Loingeville F, Nguyen TT, Riviere MK, Mentré F. Robust designs in longitudinal studies accounting for parameter and model uncertainties - application to count data. Journal of Biopharmaceutical Statistics. 2020;30(1):31-45.
[12] Seurat J, Nguyen TT, Mentré F. Robust designs accounting for model uncertainty in longitudinal studies with binary outcomes. Statistical Methods in Medical Research. 2020;29(3):934-952.
[13] Flores-Murrieta FJ, Ko HC, Flores-Acevedo DM, López-Muñoz FJ, Jusko WJ, Sale ME, Castañeda-Hernández G. Pharmacokinetic-pharmacodynamic modeling of tolmetin antinociceptive effect in the rat using an indirect response model: a population approach. J Pharmacokinet Biopharm. 1998